

Distillation for adaptation language models to Russian language

*G.K. Kovalev*¹², *M.M. Tikhomirov*¹³

© The Authors 2025. This paper is published with open access at SuperFri.org

Adapting large language models (LLMs) to morphologically rich languages like Russian presents a significant challenge, as the performance of multilingual models is often hindered by their English-centric pre-training. This paper investigates knowledge distillation (KD) as a superior method to traditional supervised fine-tuning (SFT) for the final calibration stage of language adaptation. We propose an efficient offline top- K distillation methodology, transferring knowledge from a large, Russian-adapted 32B teacher model to a smaller 4B student model by first aligning their tokenizers to enable direct logit transfer.

Our experiments demonstrate that KD consistently outperforms the SFT baseline, with an optimal configuration achieving a 4.22% performance improvement. We reveal a critical trade-off between performance and computational cost: a top-100 distillation approach yields superior gains (averaging a 3.27% increase), but at the cost of significantly higher memory requirements (62 GB vs. 7 GB for top-10). Furthermore, we find the benefits of distillation are most pronounced for student models with lower adaptive capacity (i.e., smaller LoRA α values).

Our findings offer practical guidelines, highlighting that knowledge distillation is a powerful technique for language adaptation, but its implementation must balance desired performance gains against available computational resources.

Keywords: large language model, distillation, fine-tuning, model adaptation, Russian language.

Introduction

Large Language Models (LLMs) such as Qwen [17], DeepSeek [2], Llama [3], and GPT [1] have rapidly advanced the state of the art in NLP. Despite nominal multilinguality, their pretraining data is heavily dominated by English, which limits performance in underrepresented languages. The gap is particularly pronounced for morphologically rich languages like Russian, where inflectional complexity and orthographic variation amplify subword fragmentation under default tokenizers. As a result, language adaptation - porting an existing LLM to a target language and tokenizer while preserving its instruction alignment and skills - remains a practical necessity.

A common adaptation pipeline involves extending the model’s tokenizer, performing continued pre-training, and then conducting supervised fine-tuning (SFT) to align the model with language-specific instructions. While this approach improves fluency, SFT relies on hard targets and may not be the most effective way to transfer knowledge, especially when adapting a smaller model under the guidance of a much larger, more capable one. A more potent alternative is knowledge distillation, where a “student” model learns from the full output distribution of a “teacher” model. However, distillation is often complicated by tokenizer mismatches, making direct logit transfer between different models problematic.

This paper investigates logit distillation as a superior alternative to SFT within a language adaptation pipeline. Our core methodology resolves the tokenizer mismatch problem by first aligning the vocabularies of the teacher and student models through a shared extension of Russian-specific tokens. This enables a direct and clean transfer of knowledge. Concretely, we leverage a pre-existing 32B Qwen3⁴ model, which was fully adapted for Russian. We then use it as a teacher to guide the adaptation of a 4B Qwen3⁵ student model. Our primary contribution is a comprehensive comparison showing that replacing the final SFT stage with tokenization-aligned logit distillation results in a more powerful and efficient Russian language model.

¹Lomonosov Moscow State University, Moscow, Russian Federation

²E-mail: kaengreg@ya.ru

³E-mail: tikhomirov.mm@gmail.com

⁴<https://huggingface.co/RefalMachine/RuadaptQwen3-32B-Instruct>

⁵<https://huggingface.co/Qwen/Qwen3-4B>

To ensure reproducibility and to facilitate the development of language-specific models, we publicly release our code on GitHub⁶.

1. Background

Several adaptation methods have been proposed to overcome the limitations of multilingual LLMs for languages such as Russian. The most straightforward method for such adaptation is supervised fine-tuning (SFT) on target-language instructions [14]. A more complex but efficient variant is to perform tokenization vocabulary adaptation [18] before the SFT step, which can better align the model’s internal representations with the target language’s morphology. The fine-tuning process itself is an active area of research, with methods including classical SFT, reinforcement learning from human feedback (RLHF) to align outputs with human preferences [15], and knowledge distillation, where a smaller “student” model learns from a larger “teacher” model’s outputs to transfer knowledge efficiently [8]. Among these, knowledge distillation is particularly promising for language adaptation, as it enables the creation of compact, language-specific models without full retraining. However, fine-tuning is not without challenges, as modern LLMs have dense knowledge distributions, and improper fine-tuning can lead to forgetting, where the model loses previously learned capabilities.

2. Model adaptation

Our work builds upon the methodology proposed by Tikhomirov et al. [18] for adapting large language models (LLMs) to target languages, with a focus on addressing the challenges posed by morphologically rich languages like Russian. This approach systematically modifies the model’s tokenization and internal representations to better capture language-specific nuances, followed by a calibration phase to optimize performance on target-language tasks. The adaptation process consists of the following steps:

1. **Construction of a new tokenization vocabulary:** A language-specific vocabulary is created to account for the morphological and linguistic characteristics of the target language, augmenting the original tokenizer’s vocabulary.
2. **Training embeddings for new vocabulary elements:** New token embeddings are trained to represent the added vocabulary items, ensuring compatibility with the model’s architecture and preserving semantic richness.
3. **Alignment of the base language model with the new vocabulary:** The model’s parameters are adjusted to align with the updated tokenizer, ensuring seamless integration of the new embeddings into the model’s existing knowledge [18].
4. **Transfer of core linguistic knowledge:** Core linguistic knowledge is transferred to the target version of the model using the Learned Embedding Propagation (LEP) method [18].
5. **Calibration of the adapted model:** The adapted model is fine-tuned on task-specific examples in the target language to optimize performance.

The fifth step - calibration of the adapted model - is the primary focus of our research. Calibration is critical, as it ensures the model not only understands the target language’s linguistic structures but also performs effectively on downstream tasks. To address this, we explore efficient yet computationally effective calibration methods, comparing classical supervised fine-tuning (SFT) with knowledge distillation. Classical SFT directly optimizes the model on labeled target-language data, while knowledge distillation transfers task-specific knowledge from a larger teacher model to a smaller student model.

⁶<https://github.com/RefalMachine/ruadapt>

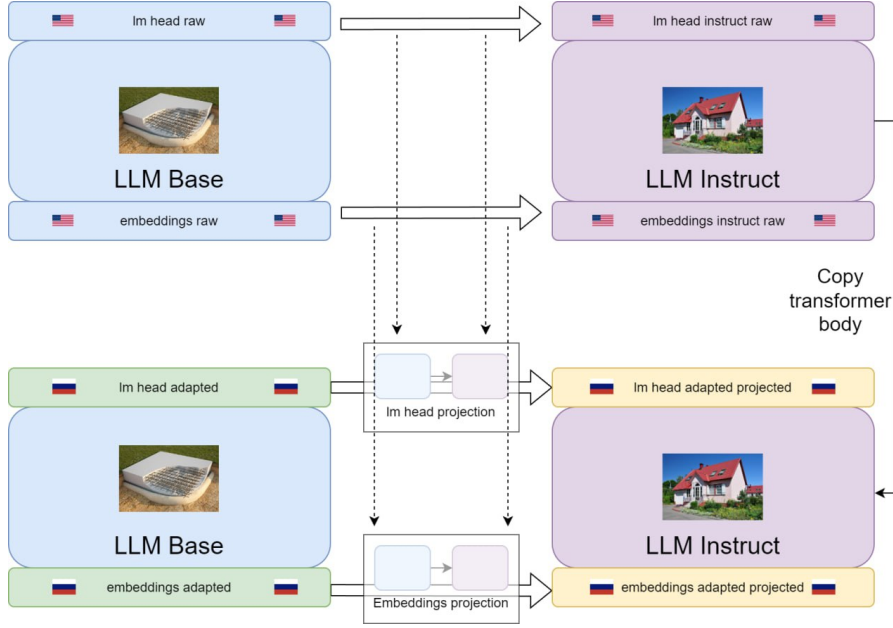


Figure 1. Language adaptation scheme

In our experiments, we selected a relatively small student model to enable extensive experimentation, thereby providing deeper insights into the benefits of our proposed methodology. Specifically, we used an adapted version of the 4B Qwen3 model, evaluating its performance on both English and Russian-specific benchmarks to assess the trade-offs between efficiency, computational cost, and task performance. Our findings aim to provide practical guidelines for adapting LLMs to low-resource languages with complex morphologies.

3. Distillation methodology

Knowledge distillation is a well-established technique for transferring knowledge from a larger “teacher” model to a smaller “student” model, enabling efficient model compression while preserving performance [7–9, 13]. Its flexibility lies not only in the variety of distillation methods but also in the ability to experiment with the teacher model’s configuration and outputs.

As previously mentioned, the student model is an adapted version of the 4B Qwen3 following the Learned Embedding Propagation (LEP) step [18]. For the “teacher” model, we chose RuadaptQwen3-32B⁷, which has been pre-adapted for Russian-language tasks and is currently the largest and most capable publicly available model using the same tokenization.

Simultaneously storing both models in memory for distillation is challenging due to their sizes, and directly training the student on the teacher’s full output distributions is computationally expensive, especially when aiming for an efficient adaptation methodology that balances quality and resource demands.

To address these challenges, we adopt a top- k offline distillation approach [16, 19], which separates the teacher logit generation and student training phases to reduce computational overhead. In the first step, we generate and store the top- k logits produced by the teacher model for each token in the training dataset, where k is a tunable hyperparameter that controls the trade-off between information retention and memory efficiency. This precomputation eliminates the need to load the teacher model during student training, significantly reducing memory requirements. In the second step, we train the student model using these precomputed logits. To further enhance efficiency, we

⁷<https://huggingface.co/RefalMachine/RuadaptQwen3-32B-Instruct>

employ parameter-efficient fine-tuning via Low-Rank Adaptation (LoRA) adapters, which minimize the number of trainable parameters while maintaining performance [10]. The student is trained on the same dataset used to generate the teacher’s logits, ensuring consistency in the knowledge transfer process.

Our training objective combines two loss functions: (1) a classical supervised fine-tuning (SFT) loss, specifically Cross-Entropy between the true tokens and the student’s predicted tokens, to calibrate the model for Russian-specific tasks; and (2) a Kullback-Leibler Divergence (KLDivLoss) term that aligns the student’s output distribution with the teacher’s precomputed top- k logits [8]. This combination enables the student model to learn both from ground-truth data and the teacher’s “dark knowledge” - patterns in the teacher’s output probabilities that enhance generalization [8]. Our combined loss function adopts the formulation proposed by Raman et al. [16], integrating Cross-Entropy and Kullback-Leibler Divergence to balance task-specific calibration and knowledge transfer from the teacher model.

$$\mathcal{L}_{\text{final}} = \mathcal{L}_{\text{CE}} + \lambda \left(1 - \exp \left(-\frac{s_i}{t_i + \varepsilon} \right) \right) \mathcal{L}_{\text{KD}} \quad (1)$$

where:

- $\mathcal{L}_{\text{CE}}$ is the cross-entropy loss between the student predictions and the ground-truth labels,
- $\mathcal{L}_{\text{KD}}$ is the knowledge distillation loss, computed as the KL divergence between the student’s and teacher’s probability distributions over the top- K tokens,
- s_i is the student logit corresponding to the ground-truth token at position i ,
- t_i is the teacher logit for the ground-truth token at position i (if the token is not in the teacher’s top- K , t_i is set to 0),
- ε is a small constant added for numerical stability,
- λ is a scaling coefficient (denoted as `loss_multi` in our implementation).

4. Data

The quality of training data is paramount in any training process, whether pre-training or fine-tuning, as it directly influences model performance. Fine-tuning, in particular, is a delicate process, as inconsistencies or contradictions between new training data and the data used for prior training can result in neutral or even negative outcomes. To ensure effective fine-tuning for Russian language adaptation, we selected the `RefalMachine/ruadapt_hybrid_instruct` dataset ⁸, which comprises approximately 80,000 instruction samples tailored for Russian-language tasks.

5. Evaluation

For evaluation in our experiments, we utilized the `llmtf` framework ⁹, an open-source toolkit designed to assess the performance of instruction-tuned language models in both few-shot and zero-shot scenarios. This framework supports flexible evaluation across diverse tasks, enabling comprehensive analysis of model capabilities on Russian-specific benchmarks.

The models were evaluated in the zero-shot setting on datasets: NEREL [11], Summ¹⁰, MultiQ, USE, taken from MERA [4], copy (generation robustness diagnostic), FLORES (ru-en and en-ru) [5], as well as enMMLU and ruMMLU, IFEval (en and ru versions) [20], ruopinione [12], ruparam [6].

⁸https://huggingface.co/datasets/RefalMachine/ruadapt_hybrid_instruct

⁹https://github.com/RefalMachine/llmtf_open

¹⁰<https://huggingface.co/datasets/IlyaGusev/gazeta>

6. Experiments

The distillation process is undoubtedly more computationally demanding. To assess its effectiveness, we conducted a series of experiments comparing classical supervised fine-tuning (SFT) with knowledge distillation during the calibration stage of model adaptation. Our central question is whether knowledge distillation can improve model performance over classical SFT, and whether the potential gains justify its additional cost.

6.1. Supervised Fine-Tuning

For the supervised fine-tuning (SFT) stage, we employed LoRA adapters [10]. Building on prior experiments, we fixed the LoRA rank at 128 and treated LoRA α together with the learning rate as tunable hyperparameters. Since our distillation approach introduces two additional hyperparameters - top- K and λ - we first identified the optimal values of LoRA α and the learning rate using classical SFT, and only then extended the setup to knowledge distillation.

Config		mean	NEREL	Summ	MultiQ	USE	flores	copy	ru babllong	en IFEval	ru IFEval	en MMLU	ru MMLU	ru opinionqa	ru param
a	lr														
64	1e-4	0.484	0.489	0.225	0.200	0.034	0.507	0.940	0.504	0.712	0.636	0.705	0.621	0.088	0.632
	2e-5	0.468	0.478	0.154	0.186	0.029	0.504	0.985	0.490	0.688	0.560	0.706	0.620	0.088	0.600
	5e-5	0.483	0.475	0.223	0.195	0.041	0.507	0.950	0.509	0.704	0.621	0.708	0.624	0.097	0.631
128	1e-4	0.483	0.495	0.224	0.194	0.037	0.508	0.945	0.504	0.717	0.601	0.706	0.620	0.092	0.634
	2e-5	0.472	0.480	0.181	0.183	0.028	0.506	0.980	0.494	0.684	0.582	0.706	0.622	0.084	0.603
	5e-5	0.482	0.478	0.223	0.193	0.031	0.508	0.955	0.507	0.693	0.608	0.705	0.620	0.091	0.653
256	1e-4	0.490	0.497	0.223	0.200	0.043	0.508	0.940	0.510	0.726	0.636	0.707	0.623	0.106	0.647
	2e-5	0.479	0.477	0.205	0.185	0.025	0.507	0.980	0.498	0.702	0.584	0.706	0.621	0.100	0.634
	5e-5	0.477	0.479	0.223	0.195	0.031	0.507	0.955	0.514	0.682	0.590	0.705	0.618	0.095	0.609
512	1e-4	0.494	0.503	0.222	0.195	0.039	0.507	0.950	0.516	0.738	0.617	0.704	0.623	0.112	0.692
	2e-5	0.475	0.475	0.218	0.189	0.028	0.506	0.970	0.509	0.710	0.584	0.707	0.620	0.088	0.568
	5e-5	0.488	0.484	0.224	0.191	0.036	0.506	0.970	0.500	0.713	0.608	0.705	0.621	0.096	0.694

Table 1. Comparison of different hyperparameter configurations (a – LoRA alpha, lr – learning rate) across multiple evaluation benchmarks.

From a performance perspective, we selected the four most promising configurations for further investigation of the distillation approach. These configurations were chosen based on their average performance across a diverse set of benchmarks, ensuring robust generalization. The selected configurations are: (1) $a = 128$, $lr=1e-4$, (2) $a = 256$, $lr=1e-4$, (3) $a = 512$, $lr=1e-4$, and (4) $a = 512$, $lr=5e-5$.

6.2. Knowledge Distillation

We adopted the LoRA rank, α , and learning rate values from the selected SFT configurations, while treating top- K and λ as the main hyperparameters of interest in this series of experiments. Each chosen SFT setup was compared against eight distillation variants, with $\lambda \in \{0.1, 0.2, 0.5, 1\}$ and top- $K \in \{10, 100\}$. The top- K parameter controls the number of top-ranked teacher model predictions used in the distillation process, while λ balances the contribution of the distillation loss against the supervised loss. To ensure a fair comparison, we maintained identical training conditions (e.g., batch size, training epochs, and dataset) across SFT and distillation experiments.

The results, presented in Table 6 (Table 7 shows more detailed results), demonstrate that knowledge distillation consistently enhances model performance over the SFT baselines. The most significant improvement was observed for the SFT baseline with $\alpha = 128$ and $lr=1e-4$. When distilled with top- $K=100$ and $\lambda = 0.5$, this model achieved an aggregate score of 0.503, a 4.22% increase over its SFT counterpart. This highlights the potential of distillation to further refine already fine-tuned models.

The Influence of top- K . A clear pattern emerges when comparing top- K values: using a larger context from the teacher model (top-100) generally yields superior results compared to a smaller one (top-10). For instance, with the $\alpha = 128$ baseline, the average improvement for top-100 variants was 3.27%, compared to 2.82% for top-10. However, this performance gain comes at a significant computational cost. Storing precomputed logits for top-100 required 62 GB of memory, whereas top-10 required only 7 GB-an 8.9-fold reduction. This trade-off makes top-10 a resource-efficient option for achieving modest gains, while top-100 is preferable when maximizing performance is the priority and resources permit.

Tuning the Distillation Weight λ . Our experiments show that $\lambda = 0.2$ emerges as a robust and generally effective choice for the distillation weight. It delivered the highest performance gains in the majority of our tested configurations. However, the single best result (4.22% growth) was achieved with $\lambda = 0.5$ in the $\alpha = 128$ setup. This suggests that while $\lambda = 0.2$ is a strong starting point for tuning, the optimal value can vary depending on other hyperparameters.

Base Config	Variant	Aggregate Score	Growth, %
SFT a=128, lr=1e-4	—	0.483	—
a=128, lr=1e-4, top-10	$\lambda=0.1$	0.498	3.17
	$\lambda=0.2$	0.497	2.95
	$\lambda=0.5$	0.495	2.56
	$\lambda=1$	0.495	2.58
Avg. Growth Top-10			2.82
Max. Growth Top-10			3.17
a=128, lr=1e-4, top-100	$\lambda=0.1$	0.498	3.20
	$\lambda=0.2$	0.498	3.15
	$\lambda=0.5$	0.503	4.22
	$\lambda=1$	0.495	2.50
Avg. Growth Top-100			3.27
Max. Growth Top-100			4.22

Table 2. SFT a=128, lr=1e-4

Base Config	Variant	Aggregate Score	Growth, %
SFT a=256, lr=1e-4	—	0.490	—
a=256, lr=1e-4, top-10	$\lambda=0.1$	0.500	2.16
	$\lambda=0.2$	0.503	2.67
	$\lambda=0.5$	0.500	2.13
	$\lambda=1$	0.500	2.08
Avg. Growth Top-10			2.26
Max. Growth Top-10			2.67
a=256, lr=1e-4, top-100	$\lambda=0.1$	0.497	1.57
	$\lambda=0.2$	0.500	2.04
	$\lambda=0.5$	0.499	1.84
	$\lambda=1$	0.496	1.33
Avg. Growth Top-100			1.70
Max. Growth Top-100			2.04

Table 3. SFT a=256, lr=1e-4

Base Config	Variant	Aggregate Score	Growth, %
SFT a=512, lr=1e-4	—	0.494	—
a=512, lr=1e-4, top-10	$\lambda=0.1$	0.496	0.38
	$\lambda=0.2$	0.502	1.63
	$\lambda=0.5$	0.496	0.41
	$\lambda=1$	0.499	1.00
Avg. Growth Top-10			0.86
Max. Growth Top-10			1.63
a=512, lr=1e-4, top-100	$\lambda=0.1$	0.502	1.66
	$\lambda=0.2$	0.503	1.83
	$\lambda=0.5$	0.501	1.46
	$\lambda=1$	0.501	1.40
Avg. Growth Top-100			1.59
Max. Growth Top-100			1.83

Table 4. SFT a=512, lr=1e-4

Base Config	Variant	Aggregate Score	Growth, %
SFT a=512, lr=5e-5	—	0.488	—
a=512, lr=5e-5, top-10	$\lambda=0.1$	0.491	0.61
	$\lambda=0.2$	0.494	1.06
	$\lambda=0.5$	0.491	0.56
	$\lambda=1$	0.489	0.18
Avg. Growth Top-10			0.60
Max. Growth Top-10			1.06
a=512, lr=5e-5, top-100	$\lambda=0.1$	0.496	1.56
	$\lambda=0.2$	0.497	1.80
	$\lambda=0.5$	0.494	1.16
	$\lambda=1$	0.488	-0.14
Avg. Growth Top-100			1.10
Max. Growth Top-100			1.80

Table 5. SFT a=512, lr=5e-5

Table 6. Combined comparison of distillation hyperparameter configurations (top- K , λ) with different SFT variants.

Conclusion

This study evaluated the effectiveness of knowledge distillation (KD) relative to classical supervised fine-tuning (SFT) during the calibration phase of large language model adaptation for Russian. Our experiments, which distilled knowledge from a Russian-adapted RuadaptQwen3-32B teacher model into a 4B Qwen3 student, conclusively demonstrate that KD is a superior approach for enhancing model performance.

A key finding from our research is the fundamental trade-off between performance and computational efficiency, governed by the top- K hyperparameter. Incorporating a broader knowledge context from the teacher (top-100) consistently yielded superior results, achieving average performance gains of up to 3.27%. This performance, however, comes at a significant cost, requiring 62 GB of memory for the precomputed logits. Conversely, a top-10 configuration emerges as a viable, resource-efficient alternative, providing modest gains with a substantially smaller memory footprint of just 7 GB.

Our analysis also provides practical tuning guidelines. We found that a distillation weight of $\lambda = 0.2$ offers a robust and effective starting point across most configurations. Furthermore, we observed that the benefits of distillation are most pronounced for models with a lower LoRA α , indicating that simpler student models with less adaptive capacity gain the most from the teacher’s guidance.

Taken together, our findings provide a practical roadmap for practitioners adapting LLMs to morphologically complex languages like Russian. They underscore the effectiveness of knowledge distillation as a powerful enhancement to the adaptation pipeline and highlight the critical need to strategically balance hyperparameter choices—such as top- K , λ , and α —against the available computational budget to achieve optimal results.

Acknowledgements

The research was supported by the Russian Science Foundation, project N^o 25-11-00191, <https://rscf.ru/project/25-11-00191/>.

The research was carried out using the MSU-270 supercomputer of Lomonosov Moscow State University.

This paper is distributed under the terms of the Creative Commons Attribution-Non Commercial 3.0 License which permits non-commercial use, reproduction and distribution of the work without further permission provided the original work is properly cited.

References

1. Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F.L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al.: Gpt-4 technical report. arXiv preprint arXiv:2303.08774 (2023)
2. DeepSeek-AI: Deepseek-v3 technical report (2024), <https://arxiv.org/abs/2412.19437>
3. Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Yang, A., Fan, A., et al.: The llama 3 herd of models. arXiv e-prints pp. arXiv–2407 (2024)
4. Fenogenova, A., Chervyakov, A., Martynov, N., Kozlova, A., Tikhonova, M., Akhmetgareeva, A., Emelyanov, A., Shevelev, D., Lebedev, P., Sinev, L., et al.: Mera: A comprehensive llm evaluation in russian. In: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 9920–9948 (2024)

5. Goyal, N., Gao, C., Chaudhary, V., Chen, P.J., Wenzek, G., Ju, D., Krishnan, S., Ranzato, M., Guzmán, F., Fan, A.: The flores-101 evaluation benchmark for low-resource and multilingual machine translation. *Transactions of the Association for Computational Linguistics* 10, 522–538 (2022)
6. Grashchenkov Pavel, V., Pasko Lada, I., Studenikina Ksenia, A., Tikhomirov Mikhail, M.: Russian parametric corpus ruparam. *Journal Scientific and Technical Of Information Technologies, Mechanics and Optics* 158(6), 991 (2024)
7. Gu, Y., Dong, L., Wei, F., Huang, M.: Minillm: Knowledge distillation of large language models. In: *The Twelfth International Conference on Learning Representations* (2023)
8. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* (2015)
9. Hsieh, C.Y., Li, C.L., Yeh, C.K., Nakhost, H., Fujii, Y., Ratner, A., Krishna, R., Lee, C.Y., Pfister, T.: Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. *arXiv preprint arXiv:2305.02301* (2023)
10. Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W.: Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685* (2021)
11. Loukachevitch, N., Artemova, E., Batura, T., Braslavski, P., Denisov, I., Ivanov, V., Manandhar, S., Pugachev, A., Tutubalina, E.: Nerel: A russian dataset with nested named entities, relations and events. In: *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*. pp. 876–885 (2021)
12. Loukachevitch, N., Moscow, L., Tkachenko, N., Lapanitsyna, A., Tikhomirov, M., Rusnachenko, N.: Ruopinionne-2024: Extraction of opinion tuples from russian news texts. In: *Proceedings of the International Conference “Dialogue. vol. 2025* (2025)
13. Men, X., Xu, M., Zhang, Q., Wang, B., Lin, H., Lu, Y., Han, X., Chen, W.: Shortgpt: Layers in large language models are more redundant than you expect. *CoRR* (2024)
14. Nikolich, A., Korolev, K., Bratchikov, S., Kiselev, I., Shelmanov, A.: Vikhr: Constructing a state-of-the-art bilingual open-source instruction-following large language model for russian. In: *Proceedings of the Fourth Workshop on Multilingual Representation Learning (MRL 2024)*. pp. 189–199 (2024)
15. Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al.: Training language models to follow instructions with human feedback. *Advances in neural information processing systems* 35, 27730–27744 (2022)
16. Raman, M., Mani, P., Liang, D., Lipton, Z.: For distillation, tokens are not all you need. In: *NeurIPS 2023 Workshop on Instruction Tuning and Instruction Following* (2023)
17. Team, Q.: Qwen3 technical report (2025), <https://arxiv.org/abs/2505.09388>
18. Tikhomirov, M., Chernyshev, D.: Facilitating large language model russian adaptation with learned embedding propagation. *Journal of Language and Education* 10(4 (40)), 130–145 (2024)
19. Zaidi, M.A., Kedia, A., Ahn, J., Kwon, T., Lee, K., Lee, H., Lee, J., et al.: Sparse logit sampling: Accelerating knowledge distillation in llms. *arXiv preprint arXiv:2503.16870* (2025)

20. Zhou, J., Lu, T., Mishra, S., Brahma, S., Basu, S., Luan, Y., Zhou, D., Hou, L.: Instruction-following evaluation for large language models. arXiv preprint arXiv:2311.07911 (2023)

Appendix

a	lr	top	λ	mean	NEREL	Summ	MultiQ	USE	flores	copy	ru babilong	en IFEval	ru IFEval	en MMLU	ru MMLU	ru opinionne	ru param
128	1e-4	10	0.1	0.498	0.485	0.224	0.219	0.089	0.508	0.950	0.508	0.721	0.628	0.708	0.624	0.107	0.704
			0.2	0.497	0.494	0.226	0.223	0.113	0.509	0.940	0.512	0.719	0.627	0.709	0.621	0.104	0.665
			0.5	0.495	0.489	0.225	0.226	0.134	0.506	0.940	0.503	0.717	0.638	0.707	0.620	0.097	0.635
			1	0.495	0.479	0.224	0.226	0.162	0.505	0.945	0.498	0.708	0.612	0.708	0.622	0.105	0.644
		100	0.1	0.498	0.482	0.226	0.222	0.116	0.510	0.950	0.517	0.713	0.623	0.708	0.623	0.102	0.685
			0.2	0.498	0.488	0.225	0.225	0.126	0.508	0.945	0.515	0.713	0.634	0.706	0.623	0.100	0.666
			0.5	0.503	0.495	0.224	0.234	0.165	0.506	0.955	0.518	0.726	0.603	0.708	0.621	0.107	0.679
			1	0.495	0.494	0.225	0.230	0.161	0.503	0.960	0.500	0.702	0.603	0.708	0.619	0.093	0.636
		256	0.1	0.502	0.487	0.224	0.225	0.087	0.508	0.945	0.512	0.750	0.632	0.706	0.619	0.107	0.725
			0.2	0.503	0.496	0.225	0.221	0.117	0.509	0.955	0.515	0.708	0.645	0.707	0.621	0.108	0.709
			0.5	0.500	0.489	0.224	0.225	0.147	0.507	0.950	0.500	0.728	0.632	0.706	0.622	0.100	0.671
			1	0.500	0.492	0.226	0.219	0.170	0.505	0.960	0.493	0.726	0.619	0.706	0.623	0.101	0.657
512	1e-4	10	0.1	0.497	0.483	0.225	0.221	0.102	0.509	0.960	0.515	0.719	0.643	0.705	0.621	0.093	0.670
			0.2	0.500	0.492	0.224	0.221	0.155	0.508	0.960	0.507	0.721	0.623	0.706	0.622	0.107	0.649
			0.5	0.499	0.498	0.225	0.218	0.164	0.506	0.960	0.490	0.710	0.630	0.708	0.621	0.111	0.642
			1	0.496	0.490	0.225	0.230	0.159	0.504	0.965	0.498	0.702	0.621	0.705	0.616	0.099	0.636
		100	0.1	0.496	0.491	0.221	0.218	0.112	0.507	0.965	0.506	0.721	0.628	0.706	0.622	0.103	0.642
			0.2	0.502	0.510	0.227	0.222	0.109	0.508	0.960	0.522	0.730	0.619	0.707	0.622	0.093	0.693
			0.5	0.496	0.498	0.225	0.219	0.121	0.506	0.960	0.511	0.726	0.614	0.707	0.625	0.113	0.619
			1	0.499	0.495	0.225	0.225	0.151	0.507	0.960	0.506	0.726	0.617	0.706	0.622	0.112	0.630
		256	0.1	0.502	0.500	0.223	0.225	0.112	0.508	0.960	0.521	0.710	0.638	0.705	0.621	0.101	0.700
			0.2	0.503	0.502	0.224	0.223	0.125	0.509	0.960	0.507	0.728	0.638	0.705	0.620	0.118	0.676
			0.5	0.501	0.498	0.226	0.225	0.151	0.507	0.965	0.491	0.721	0.628	0.706	0.621	0.113	0.659
			1	0.501	0.490	0.228	0.234	0.167	0.503	0.960	0.507	0.726	0.610	0.707	0.619	0.106	0.650
		512	0.1	0.491	0.482	0.225	0.216	0.056	0.506	0.955	0.508	0.719	0.623	0.707	0.622	0.115	0.653
			0.2	0.492	0.480	0.226	0.221	0.076	0.507	0.955	0.501	0.701	0.614	0.707	0.623	0.120	0.660
			0.5	0.491	0.468	0.228	0.220	0.104	0.507	0.960	0.506	0.715	0.628	0.706	0.620	0.104	0.618
			1	0.489	0.467	0.230	0.225	0.105	0.504	0.955	0.497	0.723	0.623	0.705	0.618	0.102	0.605
	5e-5	10	0.1	0.496	0.487	0.224	0.220	0.073	0.507	0.960	0.513	0.713	0.628	0.705	0.619	0.113	0.685
			0.2	0.497	0.486	0.226	0.225	0.093	0.507	0.955	0.502	0.721	0.634	0.706	0.618	0.118	0.671
			0.5	0.494	0.474	0.229	0.222	0.102	0.506	0.960	0.503	0.717	0.617	0.705	0.617	0.120	0.646
			1	0.488	0.484	0.229	0.231	0.112	0.503	0.965	0.486	0.717	0.595	0.706	0.617	0.093	0.601

Table 7. Comparison of different hyperparameter configurations (a – LoRA alpha, lr – learning rate) across multiple evaluation benchmarks.